Single Bit and Reduced Dimension Diffusion Strategies Over Distributed Networks

Muhammed O. Sayin, Suleyman S. Kozat*, *Senior Member, IEEE*

***Abstract*—We introduce novel diffusion based adaptive estimation strategies for distributed networks that have significantly less communication load and achieve comparable performance to the full information exchange configurations. After local estimates of the desired data is produced in each node, a single bit of information (or a reduced dimensional data vector) is generated using certain random projections of the local estimates. This newly generated data is diffused and then used in neighboring nodes to recover the original full information. We provide the complete state-space description and the mean stability analysis of our algorithms.**

***Index Terms*—Diffusion, distributed, single-bit, compressed.**

I. Introduction

DISTRIBUTED adaptive estimation utilizes a network of nodes that observe a monitored phenomena with different view points. This broadened perspective can be used to enhance estimation performance or eliminate obstructions in the environment, which may not be achieved using a single node [1]. The distributed algorithms usually target to reach the best estimate that could be produced when the individual nodes have access to all observations across the whole network. However, there is naturally a trade-off between the amount of cooperation and required communication among the nodes [1], [2].

The diffusion based distributed algorithms define a strategy in which the nodes from a predefined neighborhood could share information with each other [1], [2]. Such approaches are stable against time-varying statistical profiles [1], however, require a high amount of communication resources. For example, in a network of N nodes, where $\overline{n}$ denotes the average number of nodes in a neighborhood, then $N \times \overline{n}$ number of parameter estimates should be communicated among nodes on the average at each time.

In this letter, we propose diffusion based cooperation strategies that have significantly less communication load (e.g., a single bit of information exchange) and achieve comparable performance to the full information exchange configurations under certain settings. In this framework, after local estimates of the desired vector is produced in each node, a single bit of information (or a reduced dimensional data vector) is generated using certain random projections of the local estimates. This new information is diffused and used in neighboring nodes instead of the original estimates; significantly reducing the communication load in the network. We only require synchronization of this randomized projection operation, which can be achieved using simple pilot signals [3]. Note that our approach differs from quantization based diffusion strategies such as [4] in terms of the compression of the diffused information. In [4], a quantized parameter estimate is exchanged among nodes to avoid infinite precision in the transmission. In [5], the sign of the innovation sequence in the context of decentralized estimation and in [6], the relative difference between the states of the nodes in a consensus network are exchanged using a single bit of information. Here, we substantially compress the exchanged information, even to a single bit, and perform local adaptive operations at each node to recover the full information vector. In this sense, our method is more akin to compressive sensing rather than to a quantization framework.

Our main contributions include: 1) We propose algorithms to significantly reduce the amount of communication between nodes for diffusion based distributed strategies; 2) We analyze the stability of the algorithms in the mean under certain statistical conditions; 3) We illustrate the comparable convergence performance of these algorithms in different numerical examples. We emphasize that although we only provide the mean stability analysis due to space limitations, the mean-square convergence and tracking analysis are carried out in a similar fashion (following [1]) in a separate paper submission.

The letter is organized as follows. In Section II, we introduce the framework and the studied problem. The new approaches are derived in Section III. In Section IV, we analyze the mean stability of our approaches. Numerical examples and concluding remarks are provided in Section V.

II. Problem Description

Consider the widely studied spatially distributed framework [1], [2]. Here, we have N number of nodes where two nodes are considered neighbors if they can exchange information. For a node i, the set of indexes of its neighbors including the index of itself is denoted by $\mathcal{N}_i$. At each node, an unknown desired vector[1], $\boldsymbol{w}_o \in \mathbb{R}^m$, is observed through a linear model $d_i(t) = \boldsymbol{w}_o^T \boldsymbol{u}_i(t) + v_i(t)$,[2] assuming the observation noise is temporally and spatially white (or independent), i.e., $E[v_i(t)v_j(l)] = \sigma_i^2 \delta(i-j)\delta(t-l)$, where $\delta(\cdot)$ is the Kronecker delta and σ_i^2 is the variance of the noise. The regression vectors

This work is supported in part by TUBITAK, Contract No:112E161.
Muhammed O. Sayin (m_sayin@ug.bilkent.edu.tr) and Suleyman S. Kozat (kozat@ee.bilkent.edu.tr) are with the Department of Electrical and Electronics Engineering, Bilkent University, Ankara, Turkey (Tel: +90312 2902336 and Fax: +90312 2901223)

[1]Although we assume a time invariant desired vector, our derivations can be readily extended to certain non-stationary models [3].

[2]We represent vectors (matrices) by bold lower (upper) case letters. For a matrix $\boldsymbol{A}$ (or a vector $\boldsymbol{a}$), $\boldsymbol{A}^T$ is the transpose. $\|\boldsymbol{a}\|$ is the Euclidean norm. For notational simplicity we work with real data and all random variables have zero mean. The sign of a is denoted by $\mathrm{sign}(a)$ (0 is considered positive without loss of generality). For a vector $\boldsymbol{a}$, $\dim(\boldsymbol{a})$ denotes the length. The expectation of a vector or a matrix is denoted with an over-line, i.e. $E[\boldsymbol{a}] = \overline{\boldsymbol{a}}$. The $\mathrm{diag}(\boldsymbol{A})$ returns a new matrix with only the main diagonal of $\boldsymbol{A}$ while $\mathrm{diag}(\boldsymbol{a})$ puts $\boldsymbol{a}$ on the main diagonal of the new matrix.

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

are also assumed to be spatially and temporally uncorrelated with each other and with the observation noise. At each node an adaptive estimation algorithm is working such as the LMS algorithm [3] given as

$$\boldsymbol{\phi}_i(t+1) = (\boldsymbol{I} - \mu_i \boldsymbol{u}_i(t)\boldsymbol{u}_i^T(t))\boldsymbol{w}_i(t) + \mu_i d_i(t)\boldsymbol{u}_i(t),$$

$\mu_i > 0$. As the diffusion strategy, we use the adapt-then-combine (ATC) diffusion strategy as an example since it is shown to outperform the combine-then-adapt diffusion and consensus strategies under certain conditions [7]. However, our derivations also cover these distributed strategies. In the ATC strategy, at each node i, the final estimate is constructed as

$$\boldsymbol{w}_i(t+1) = \sum_{k \in \mathcal{N}_i} \lambda_{i,k} \boldsymbol{\phi}_k(t+1),$$

where $\lambda_{i,k}$'s are the combination weights $\sum_{k\in\mathcal{N}_i} \lambda_{i,k} = 1$ and $\lambda_{i,k} \geq 0$. The combination weights $\lambda_{i,k}$ can also be adapted in time, affinely constrained or unconstrained [8]. We stick to constant-in-time weights with the simplex constraint since the stabilization effect of such weights is demonstrated in [1].

In the diffusion based distributed networks, whole parameter estimates are exchanged within the neighborhood. In the next section, we introduce two different approaches in order to reduce the amount of information exchange between nodes.

III. New Diffusion Strategies

A. Reduced Dimension Diffusion

In the first approach, each node calculates a reduced dimensional vector through a linear transformation $\boldsymbol{z}_k(t+1) = \boldsymbol{C}(t+1)\boldsymbol{\phi}_k(t+1)$, where $\dim(\boldsymbol{z}_k(t+1)) \ll \dim(\boldsymbol{\phi}_k(t+1))$, and transmits $\boldsymbol{z}_k(t+1)$ instead of $\boldsymbol{\phi}_k(t+1)$. We use a randomized linear transformation matrix $\boldsymbol{C}(t+1)$ where the size of the matrix determines the compression amount. Each neighboring node uses the same $\boldsymbol{C}(t+1)$. After receiving $\boldsymbol{z}_k(t+1)$, a neighbor node i constructs an estimate $\boldsymbol{a}_k(t+1)$ of the original $\boldsymbol{\phi}_k(t+1)$ using a *minimum disturbance criteria* [3] as

$$\begin{aligned} &\boldsymbol{a}_k(t+1) = \arg\min_{\boldsymbol{a}} \|\boldsymbol{a} - \boldsymbol{a}_k(t)\| \qquad (1)\\ &\text{such that } \ \boldsymbol{C}(t+1)\boldsymbol{a} = \boldsymbol{z}_k(t+1), \end{aligned}$$

where $\boldsymbol{a}_k(t) \in \mathbb{R}^m$. Note that (1) yields the NLMS algorithm [3] as

$$\begin{aligned} \boldsymbol{a}_k(t+1) = \boldsymbol{a}_k(t) + \sigma_k \boldsymbol{C}(t+1)^T \big[\boldsymbol{C}(t+1)\times \\ \boldsymbol{C}(t+1)^T\big]^{-1}\big(\boldsymbol{z}_k(t+1) - \boldsymbol{C}(t+1)\boldsymbol{a}_k(t)\big), \end{aligned} \qquad (2)$$

where a learning rate $\sigma_k > 0$. is also incorporated after (1). After $\boldsymbol{a}_k(t+1)$'s are calculated, we construct the final estimate at node i as

$$\boldsymbol{w}_i(t+1) = \lambda_{i,i}\boldsymbol{\phi}_i(t+1) + \sum_{k\in\mathcal{N}_i\setminus i} \lambda_{i,k}\boldsymbol{a}_k(t+1). \qquad (3)$$

Remark 1: For a time invariant projection matrix, $\boldsymbol{C}(t) = \boldsymbol{C}$, the exchanged estimate $\boldsymbol{a}_k(t)$ converges to the projection of the original parameter estimate $\boldsymbol{\phi}_k(t)$ onto the column space of the matrix $\boldsymbol{C}$ (provided that adaptation is fast enough). In order to avoid biased convergence, we choose randomized projection matrices that span the whole parameter space.

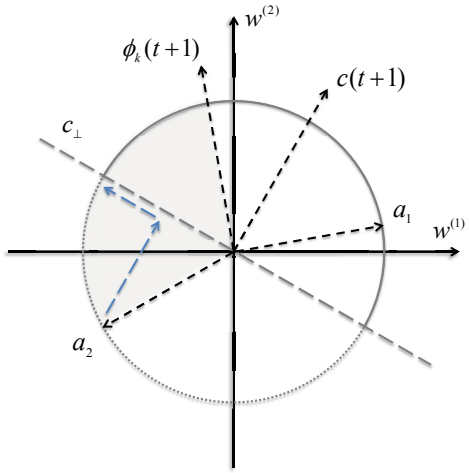

Fig. 1: Single bit diffusion in two dimensions, i.e., $\boldsymbol{w}_o \in \mathbb{R}^2$. As an example, one can have $\boldsymbol{a}_k(t) = \boldsymbol{a}_1$ or $\boldsymbol{a}_k(t) = \boldsymbol{a}_2$. $\boldsymbol{c}_\perp$ denotes the vector space perpendicular to $\boldsymbol{c}(t+1)$ in two dimensions and the shaded area represents the update region for $\boldsymbol{a}_k(t) = \boldsymbol{a}_2$

Remark 2: One can also use an ordinary LMS update to train $\boldsymbol{a}_k(t)$ to avoid the inversion operation in (2), considering $\boldsymbol{z}_k(t+1)$ as the desired data and $\boldsymbol{C}(t+1)$ as the regression matrix. However, since the dimensions of $\boldsymbol{z}_k(\cdot)$'s are much smaller than the dimension of $\boldsymbol{w}_o$, e.g., in our simulations we use scalar $z_k(\cdot)$'s with $m = 1$, one can use the NLMS update for $\boldsymbol{a}_k(\cdot)$'s without significant computational increase.

In the following, we further reduce the amount of transmitted information by diffusing a single bit of information instead of a scalar.

B. Single Bit Diffusion

In this approach, we exchange only the sign of the linear transformation $z_k(t+1) = \boldsymbol{c}(t+1)^T\boldsymbol{\phi}_k(t+1)$. According to the transmitted sign, the neighboring node i can construct an estimate $\boldsymbol{a}_k(t+1)$ of $\boldsymbol{\phi}_k(t+1)$ as

$$\boldsymbol{a}_k(t+1) = \arg\min_{\boldsymbol{a}} \|\boldsymbol{a} - \boldsymbol{a}_k(t)\| \text{ such that} \qquad (4)$$

$$\text{sign}\left(\boldsymbol{c}(t+1)^T\boldsymbol{a}\right) = \text{sign}\left(z_k(t+1)\right) \text{ and} \qquad (5)$$

$$\|\boldsymbol{a}\| = 1. \qquad (6)$$

To solve (4), we observe from Fig. 1 that we can only have two different cases for $\boldsymbol{a}_k(t)$. In the first case, we have $\text{sign}\left(\boldsymbol{c}(t+1)^T\boldsymbol{a}_k(t)\right) = \text{sign}\left(z_k(t+1)\right)$, e.g., $\boldsymbol{a}_k(t) = \boldsymbol{a}_1$ in the figure. In this case, no update is needed, $\boldsymbol{a}_k(t+1) = \boldsymbol{a}_k(t)$, since $\boldsymbol{a}_k(t)$ satisfies both conditions (5), (6) and $\|\boldsymbol{a}_k(t+1) - \boldsymbol{a}_k(t)\| = 0$. In the second case, we have $\text{sign}\left(\boldsymbol{c}(t+1)^T\boldsymbol{a}_k(t)\right) \neq \text{sign}\left(z_k(t+1)\right)$, e.g., $\boldsymbol{a}_k(t) = \boldsymbol{a}_2$ in the figure. For this case, i.e., $\boldsymbol{a}_k(t) = \boldsymbol{a}_2$, we only need to project $\boldsymbol{a}_k(t)$ to the half hyper sphere (shown as a half circle in two dimensions in Fig. 1), which corresponds to the constraints (5) and (6). This projection can be readily accomplished by first projecting $\boldsymbol{a}_k(t)$ to the vector space perpendicular to $\boldsymbol{c}(t+1)$ and then scaling the projected vector to have unit norm. This yields the following

$$\boldsymbol{a}_k(t+1) = \frac{\boldsymbol{a}_k(t) - \gamma(t+1)\boldsymbol{c}(t+1)\boldsymbol{c}(t+1)^T\boldsymbol{a}_k(t)}{\|\boldsymbol{a}_k(t) - \gamma(t+1)\boldsymbol{c}(t+1)\boldsymbol{c}(t+1)^T\boldsymbol{a}_k(t)\|}$$

update, where

$$\gamma(t+1) \triangleq \frac{1-\text{sign}\left(z_k(t+1)\right)\text{sign}\left(\boldsymbol{c}(t+1)^T\boldsymbol{a}_k(t)\right)}{2\;\boldsymbol{c}(t+1)^T\boldsymbol{c}(t+1)}.$$

Here, (6) is needed to resolve the inherent amplitude uncertainty in (5) since the diffused sign bit does not carry any amplitude information. Without (6), the amplitude of the constrained estimates diminishes to zero by updates. We resolve the amplitude uncertainty in the final combination by multiplying the unit norm estimate $\boldsymbol{a}_k(t+1)$ with the magnitude of the local parameter estimate $\boldsymbol{\phi}_i(t+1)$. This scaling with the norm of $\boldsymbol{\phi}_i(t+1)$ results in the rotated parameter estimation in the direction of $\boldsymbol{a}_k(t+1)$. After the construction of the exchange estimates, the final estimate $\boldsymbol{w}_i(t+1)$ is calculated as

$$\boldsymbol{w}_i(t+1) = \lambda_{i,i}\boldsymbol{\phi}_i(t+1) + \|\boldsymbol{\phi}_i(t+1)\| \sum_{k\in\mathcal{N}_i\setminus i} \lambda_{i,k}\boldsymbol{a}_k(t+1).$$

Remark 3: Fig. 1 also demonstrates the update procedure for $\boldsymbol{a}_k(t) = \boldsymbol{a}_2$. The update is performed if the line, $\boldsymbol{c}_\perp$, perpendicular to $\boldsymbol{c}(t+1)$ passes through the shaded update region. Otherwise, the exchanged sign provides no new information and is discarded.

Alternatively, we can also resolve the amplitude uncertainty by using a sign LMS [3] based approach. In this approach, at each node, we run an adaptive algorithm considering $\boldsymbol{c}(t+1)^T\boldsymbol{\phi}_k(t+1)$ as the desired data and $\boldsymbol{c}(t+1)$ as the regression vector. We then diffuse the sign of the error $z_k(t+1) = \epsilon_k(t+1) \triangleq \boldsymbol{c}(t+1)^T\boldsymbol{\phi}_k(t+1) - \boldsymbol{c}(t+1)^T\boldsymbol{a}_k(t)$. Using the sign algorithm [3], each node k can construct the exchange estimate as

$$\boldsymbol{a}_k(t+1) = \boldsymbol{a}_k(t) + \sigma_k\;\text{sign}(z_k(t+1))\boldsymbol{c}(t+1). \quad (7)$$

Assuming $\boldsymbol{a}_k(t)$'s are initialized with the same values at each node, (7) can be repeated at all neighboring nodes of k to produce the same $\boldsymbol{a}_k(t)$. In the next section, we analyze the global stability of the algorithms in the mean.

IV. Stability Analysis

We can write the reduced dimension diffusion (2) and the sign algorithm inspired diffusion (7) approaches in a compact form as

$$\boldsymbol{\phi}_i(t+1) = \boldsymbol{w}_i(t) + \mu_i\boldsymbol{u}_i(t)e_i(t), \quad (8)$$

$$\boldsymbol{a}_k(t+1) = \boldsymbol{a}_k(t) + \sigma_k\boldsymbol{c}(t+1)h\left(\epsilon_k(t+1),\boldsymbol{c}(t+1)\right) \quad (9)$$

$$\boldsymbol{w}_i(t+1) = g_i\left(\boldsymbol{\phi}_i(t+1),\boldsymbol{a}_k(t+1); k\in\mathcal{N}_i\setminus i\right), \quad (10)$$

where $\mu_i > 0$ and $\sigma_k > 0$ are the local learning rates, $g_i(\cdot)$ is a combination function such as (3) and

$$e_i(t) = d_i(t) - \boldsymbol{u}_i(t)^T\boldsymbol{w}_i(t),$$
$$\epsilon_k(t+1) = \boldsymbol{c}(t+1)^T\left(\boldsymbol{\phi}_k(t+1) - \boldsymbol{a}_k(t)\right)$$

are the estimation and projected reconstruction errors. Here, $h\left(\epsilon_k(t+1),\boldsymbol{c}(t+1)\right)$ is a generic function of $\epsilon_k(t+1)$ and $\boldsymbol{c}(t+1)$, e.g., for the scalar diffusion case $h\left(\epsilon_k(t+1),\boldsymbol{c}(t+1)\right) = \left(\boldsymbol{c}(t+1)^T\boldsymbol{c}(t+1)\right)^{-1}\epsilon_k(t+1)$.

We define deviations from the parameter of interests as

$$\Delta\boldsymbol{\phi}_k(t+1) = \boldsymbol{w}_o - \boldsymbol{\phi}_k(t+1), \quad (11)$$

$$\Delta\boldsymbol{a}_k(t+1) = \boldsymbol{\phi}_k(t+1) - \boldsymbol{a}_k(t+1). \quad (12)$$

Substituting (12) into (10), we get the final estimate as

$$\boldsymbol{w}_i(t+1) = \sum_{k\in\mathcal{N}_i}\lambda_{i,k}\boldsymbol{\phi}_k(t+1) - \sum_{k\in\mathcal{N}_i\setminus i}\lambda_{i,k}\;\Delta\boldsymbol{a}_k(t+1). \quad (13)$$

In the analysis of the mean stability, we make the following assumptions:

1) The projection signal $\boldsymbol{c}(t)$ (or $\boldsymbol{C}(t)$) and the regression data $\boldsymbol{u}_k(t)$ are temporally independent.

2) The *a priori* construction error $\epsilon_k(\cdot)$ and the projection signal $\boldsymbol{c}(\cdot)$ (or $\boldsymbol{C}(\cdot)$) are jointly Gaussian. For sufficiently small step size and long filter length, this assumption is true [3].

3) The original parameter estimates $\boldsymbol{\phi}_i(\cdot)$ vary slowly relative to the constructed estimates $\boldsymbol{a}_i(\cdot)$ through the appropriate step sizes such that

$$\Delta\boldsymbol{a}_k(t) = \boldsymbol{\phi}_k(t) - \boldsymbol{a}_k(t) \cong \boldsymbol{\phi}_k(t+1) - \boldsymbol{a}_k(t) \text{ or}$$
$$\Delta\boldsymbol{a}_k(t+1) = \boldsymbol{\phi}_k(t+1) - \boldsymbol{a}_k(t+1) \cong \boldsymbol{\phi}_k(t) - \boldsymbol{a}_k(t+1).$$

We then define the following global variables

$$\Delta\boldsymbol{\phi}(t) \triangleq \begin{bmatrix}\Delta\boldsymbol{\phi}_1(t)\\ \vdots \\ \Delta\boldsymbol{\phi}_N(t)\end{bmatrix},\;\; \Delta\boldsymbol{a}(t) \triangleq \begin{bmatrix}\Delta\boldsymbol{a}_1(t)\\ \vdots \\ \Delta\boldsymbol{a}_N(t)\end{bmatrix},$$

$$\boldsymbol{U}(t) \triangleq \begin{bmatrix}\boldsymbol{u}_1(t) & \dots & \boldsymbol{0}\\ \vdots & \ddots & \vdots \\ \boldsymbol{0} & \dots & \boldsymbol{u}_N(t)\end{bmatrix},\;\; \boldsymbol{v}(t) \triangleq \begin{bmatrix}v_1(t)\\ \vdots \\ v_N(t)\end{bmatrix},$$

where the vector dimensions are $(mN\times 1)$ and the matrix dimensions are $(mN\times N)$.

Using (8), (9), (11), (12), and (13), we get

$$\Delta\boldsymbol{\phi}(t+1) = \left(\boldsymbol{I} - \boldsymbol{D}\boldsymbol{U}(t)\boldsymbol{U}(t)^T\right)\boldsymbol{G}\;\Delta\boldsymbol{\phi}(t) - \left(\boldsymbol{I} - \boldsymbol{D}\boldsymbol{U}(t)\boldsymbol{U}(t)^T\right)\tilde{\boldsymbol{G}}\;\Delta\boldsymbol{a}(t) + \boldsymbol{D}\boldsymbol{U}(t)\boldsymbol{v}(t), \quad (14)$$

where $\boldsymbol{G} \triangleq \boldsymbol{\Lambda}\otimes\boldsymbol{I}_m$ is the transition matrix (and $\otimes$ is the Kronecker product), $\tilde{\boldsymbol{G}} \triangleq \boldsymbol{G} - \text{diag}\left(\boldsymbol{G}\right)$, $\boldsymbol{\Lambda} \triangleq [\lambda_{i,k}]$ is the combination matrix and $\boldsymbol{D} \triangleq \text{diag}\left([\mu_1,\mu_2,...,\mu_N]\right)\otimes\boldsymbol{I}_m$.

The global update for the reconstructed parameters yields

$$\Delta\boldsymbol{a}(t+1) = \left(\boldsymbol{I} - \boldsymbol{S}\boldsymbol{H}(t)\right)\Delta\boldsymbol{a}(t), \quad (15)$$

where $\boldsymbol{S} \triangleq \text{diag}\left([\sigma_1,\sigma_2,...,\sigma_N]\right)\otimes\boldsymbol{I}_m$ and $\boldsymbol{H}(t)$ is an appropriate transition matrix. As an example, for the scalar diffusion case we have

$$\boldsymbol{H}(t) = \boldsymbol{I}_m\otimes\left(\frac{\boldsymbol{c}(t+1)\boldsymbol{c}(t+1)^T}{\boldsymbol{c}(t+1)^T\boldsymbol{c}(t+1)}\right).$$

For the single-bit diffusion, $h(\epsilon_k(t+1),\boldsymbol{c}(t+1)) = \text{sign}(\epsilon_k(t+1))$ is a nonlinear function of $\epsilon_k(t+1)$, hence it is not straightforward to write (15). Although $h(\epsilon_k(t+1),\boldsymbol{c}(t+1))$ is nonlinear, it can be linearized using a Taylor series expansion. However, since we assume joint Gaussianity, by the Price's theorem [3], we can write the expectation of the deviation $\Delta\boldsymbol{a}_k(t+1)$ as [3]

$$\Delta\overline{\boldsymbol{a}}_k(t+1) = \Delta\overline{\boldsymbol{a}}_k(t) - \sigma_k\sqrt{\frac{2}{\pi}}\frac{E\left[\boldsymbol{c}(t+1)\boldsymbol{c}(t+1)^T\right]}{E\left[\epsilon_k^2(t+1)\right]}\Delta\overline{\boldsymbol{a}}_k(t).$$

Note that the variance $E\left[\epsilon_k^2(t+1)\right]$ is given by $E\left[\left(\boldsymbol{c}(t+1)\Delta\overline{\boldsymbol{a}}(t)\right)^2\right]$, and we define $\boldsymbol{F}(t+1) \triangleq \sqrt{\frac{\pi}{2}}\;\text{diag}\left([\epsilon_1^2(t+1),...,\epsilon_N^2(t+1)]\right)$ that leads

$$\overline{\boldsymbol{H}}(t) = \left[\overline{\boldsymbol{F}}(t+1)\otimes\boldsymbol{I}_m\right]^{-1}\left[\boldsymbol{I}_m\otimes\left(E\left[\boldsymbol{c}(t+1)\boldsymbol{c}(t+1)^T\right]\right)\right].$$

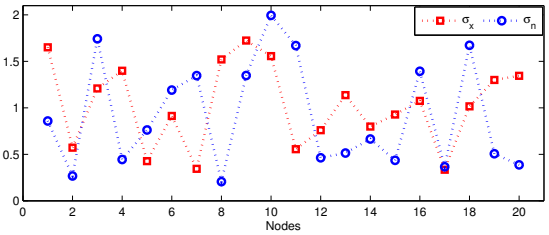

Fig. 2: Statistical profile of the example network.

Taking the expectation of both sides of (14) and (15) and by the first assumption, we get

$$\begin{bmatrix} \Delta\overline{\boldsymbol{\phi}}(t+1) \\ \Delta\overline{\boldsymbol{a}}(t+1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{B}\boldsymbol{G} & \boldsymbol{B}\tilde{\boldsymbol{G}} \\ \boldsymbol{0} & \boldsymbol{I} - \boldsymbol{S}\overline{\boldsymbol{H}}(t) \end{bmatrix} \begin{bmatrix} \Delta\overline{\boldsymbol{\phi}}(t) \\ \Delta\overline{\boldsymbol{a}}(t) \end{bmatrix}, \qquad (16)$$

where $\boldsymbol{B} \triangleq \boldsymbol{I} - \boldsymbol{D}\overline{\boldsymbol{U}(t)\boldsymbol{U}(t)^T}$. (16) covers both the reduced dimension and single bit diffusion strategies. From (16) we also observe that our algorithms are stable in the mean if $|\lambda\left(\boldsymbol{I} - \boldsymbol{S}\overline{\boldsymbol{H}}\right)| < 1$ (provided that the full diffusion scheme is stable), where $\lambda(\cdot)$'s are the eigenvalues. As example, for the scalar case, assuming $\boldsymbol{c}(\cdot)$ are i.i.d. zero mean with unit variance, then $|\lambda\left(\boldsymbol{I} - \boldsymbol{S}\overline{\boldsymbol{H}}\right)| < 1$ if and only if $|1 - \sigma_i| < 1$ for all i. Furthermore, the step sizes σ_i for the reconstruction algorithms could be chosen accordingly for comparable convergence performance with the full diffusion case. Following examples illustrate these results.

V. Numerical Example and Concluding Remarks

In this section, we compare the introduced algorithms with the full diffusion and no-cooperation schemes for the example network with $N = 20$ nodes. Here, we have stationary data $d_i(t) = \boldsymbol{w}_o^T \boldsymbol{u}_i(t) + v_i(t)$ for $i = 1, 2, ..., N$, where $\boldsymbol{u}_i(t)$ and $v_i(t)$ are i.i.d. zero mean and their variances and $\boldsymbol{w}_o \in \mathbb{R}^4$ are randomly chosen (See Fig. 2).

The combination matrix $\boldsymbol{\Lambda} = [\lambda_{i,k}]$ is chosen as

$$\lambda_{i,k} = \begin{cases} 1/\max(n_i, n_k) & \text{if } i \neq k \text{ are linked,} \\ 0 & \text{for } i \text{ and } k \text{ not linked,} \\ 1 - \sum_{k \in \mathcal{N}_i \setminus i} \lambda_{i,k} & \text{for } i = k, \end{cases}$$

where n_i and n_k denote the number of neighboring nodes for i and k according to the Metropolis rule.

The step sizes for the adaptation algorithms (8) of diffusion schemes are set such that they converge with the same rate: 0.1 for no-cooperation scheme (i.e. the combination matrix $\boldsymbol{\Lambda} = \boldsymbol{I}_N$), 0.2 for the single-bit and reduced dimension diffusion strategies, and 0.028 for the full diffusion configuration. The step sizes σ_i for the reconstruction algorithms (9) are set as 0.25, 0.72, 0.36, and 0.18 for the single-bit, one-dimension, two-dimension, and three-dimension, respectively. The randomized projection vectors $\boldsymbol{c}(t)$ (and matrices $\boldsymbol{C}(t)$) are generated i.i.d. zero mean Gaussian with standard deviation 1. We point out that we set the learning rates for all algorithms such that the convergence rate of all algorithms are the same for a fair comparison of the final MSDs.

In Fig. 3, we compare the mean-square deviation of various diffusion schemes in terms of their steady state MSDs for the same convergence rate. As expected, in our simulations, the introduced algorithms readily outperform the no-cooperation scheme in terms of the final MSDs. We observe from these

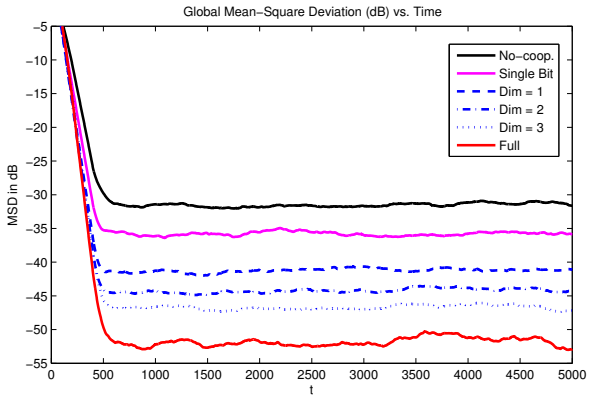

Fig. 3: Global mean-square deviation (MSD) of diffusion and no-cooperation schemes.

simulations that although we significantly reduce the amount of information exchange, the introduced algorithms perform similar to the full information case. To illustrate this further, in Fig. 3, we plot the performance of the reduced-dimension algorithm where we gradually increase the number of dimensions that we kept. We observe that as the number of dimensions increases, the reduced-dimension algorithm gradually achieves the performance of the full information case. Note also that the no-cooperation scheme gives stable error because the adaptation algorithms converge at all nodes. If at least one of the nodes diverges, then the performance of the no-cooperation scheme degrades severely whereas this does not usually influence the diffusion algorithms.

In this letter we introduce novel diffusion based distributed adaptive estimation algorithms that significantly reduce the communication load while providing comparable performance with the full information exchange approaches in our simulations. We achieve this by exchanging either a scalar or a single bit of information generated from random projections of the estimated vectors at each node. Based on these exchanged information, each node recalculates the estimates generated by its neighboring nodes (which are then subsequently merged). We also provide a mean stability analysis of the introduced approaches for stationary data. This analysis can also be extended to mean-square and tracking analysis under certain settings.

References

[1] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, 2008.

[2] F. S. Cattivelli and A. H. Sayed, "Diffusion lms strategies for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1035–1048, 2010.

[3] A. H. Sayed, *Fundamentals of Adaptive Filtering*. John Wiley and Sons, 2003.

[4] S. Xie and H. Li, "Distributed LMS estimation over networks with quantised communications," *International Journal of Control*, vol. 86, no. 3, pp. 478–492, 2013.

[5] A. Ribeiro, G. Giannakis, and S. Roumeliotis, "Soi-kf: Distributed kalman filtering with low-cost communications using the sign of innovations," *Signal Processing, IEEE Transactions on*, vol. 54, no. 12, pp. 4782–4795, 2006.

[6] H. Sayyadi and M. R. Doostmohammadian, "Finite-time consensus in directed switching network topologies and time-delayed communications," *Scientia Iranica*, vol. 18, no. 1, pp. 75–85, February 2011.

[7] S. Y. Tu and A. H. Sayed, "On the influence of informed agents on learning and adaptation over networks," *IEEE Transactions on Signal Processing*, vol. 61, no. 6, pp. 1339–1356, March 2013.

[8] S. S. Kozat, A. T. Erdogan, A. C. Singer, and A. H. Sayed, "Steady state MSE performance analysis of mixture approaches to adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 58, no. 8, pp. 4050–4063, August 2010.